Deep Learning

Andrea Asperti

DISI - Department of Informatics: Science and Engineering
University of Bologna
Mura Anteo Zamboni 7, 40127, Bologna, ITALY
andrea.asperti@unibo.it

Deep Learning in a nutshell

Q: **What is Deep Learning?**

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

A: All problems involving myriads of features, e.g. images/speech/text processing, recommendation systems, ...

Deep Learning in a nutshell

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

A: All problems involving myriads of features, e.g. images/speech/text processing, recommendation systems, . . .

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why "Deep"?**

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why "Deep"?**

A:
- because it exploits Deep Neural Networks, composed by many nested layers of neurons

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why "Deep"?**

A:
- because it exploits Deep Neural Networks, composed by many nested layers of neurons
- because it exploits deep features of data, that is features extracted from other features

ML recap

What is Machine Learning about?

There are problems that are difficult to address with traditional programming techniques:

- classify a document according to some criteria (e.g. spam, sentiment analysis, ...)
- compute the probability that a credit card transaction is fraudulent
- recognize an object in some image (possibly from an inusual viewpoint, in new lighting conditions, in a cluttered scene)
- ...

Typically the result is a weighted combination of a large number of parameters, each one contributing to the solution in a small degree.

The Machine Learning approach

Suppose to have a set of input-output pairs (training set)

$$\{\langle x_i, y_i \rangle\}$$

the problem consists in guessing the map $x_i \mapsto y_i$

The M.L. approach:

- describe the problem with a **model** depending on some parameters Θ (i.e. choose a parametric class of functions)
- define a **loss function** to compare the results of the model with the expected (experimental) values
- **optimize** (fit) the parameters Θ to reduce the loss to a minimum

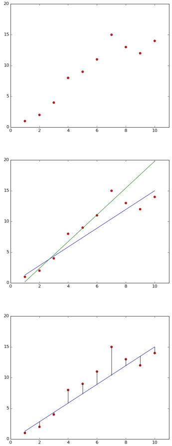
Example: a regression problem

You have some points on the plane and you want to fit a line through them

Step 1 Fix a parametric class of models.
For intance linear functions $y = ax + b$;
a and b are the parameters of the model

Step 2 Fix a way to decide when a line is better than another (loss function)
For instance, mean square error (mse)

Step 3 Try to tune the parameters in order to reduce the loss (training).



Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

So, we use **iterative** techniques (typically, gradient descent) to progressively approximate the result.

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

So, we use **iterative** techniques (typically, gradient descent) to progressively approximate the result.

This form of iteration over data can be understood as a way of progressive learning of the objective function based on the experience of past observations.

Using gradients

Goal: minimize a loss function E over (fixed) training samples:

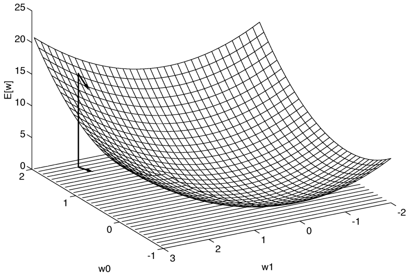
$$\Theta(w) = \sum_i E(o(w, x_i), y_i)$$

See how it changes according to small perturbations $\Delta(w)$ of the parameters w: this is the gradient

$$\nabla_w[\theta] = [\frac{\partial\Theta}{\partial w_1}, \ldots, \frac{\partial\Theta}{\partial w_n}]$$

of Θ w.r.t. w.

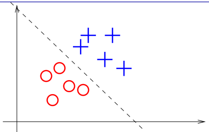
The gradient is a vector pointing in the direction of steepest ascent.



A bit of taxonomy

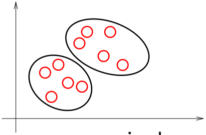
Different types of Learning Tasks

- **supervised learning**:
  inputs + outputs (labels)
  - classification
  - regression



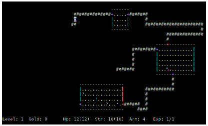
supervised

- **unsupervised learning**:
  just inputs
  - clustering
  - component analysis
  - anomaly detection
  - autoencoding



unsupervised

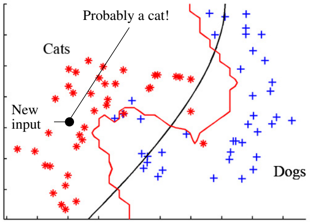
- **reinforcement learning**
  actions and rewards
  - learning long-term gains
  - planning



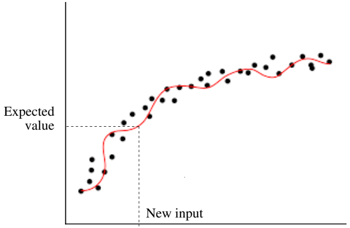
reinforcement

Classification vs. Regression

Two forms of supervised learning: $\{\langle x_i, y_i \rangle\}$



classification



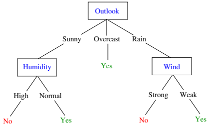
regression

y is discete: $y \in \{\bullet, +\}$

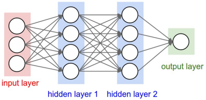
y is (conceptually) continuous

Many different techniques

- **Different ways to define the models**:
  - decision trees
  - linear models
  - neural networks
  - ...

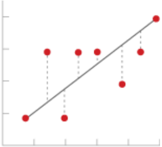


decision tree

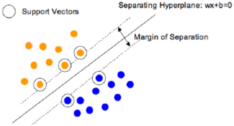


neural net

- **Different error (loss) functions**:
  - mean squared errors
  - logistic loss
  - cross entropy
  - cosine distance
  - maximum margin
  - ...



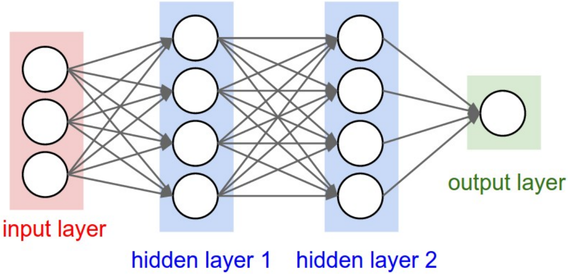
mean squared errors



maximum margin

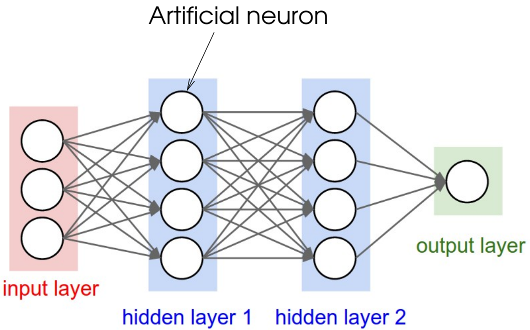
Next argument

Neural Networks



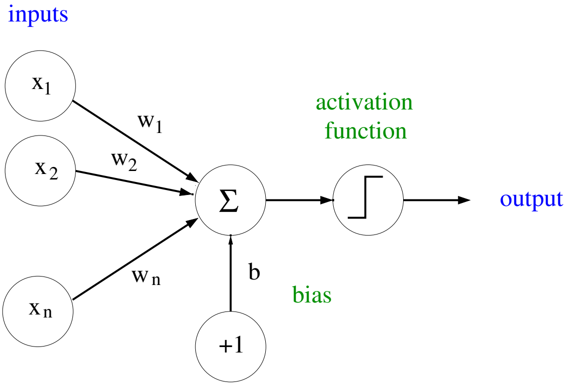
Neural Network

A network of (artificial) neurons



Each neuron takes multiple inputs and produces a single output (that can be passed as input to many other neurons).

The artificial neuron



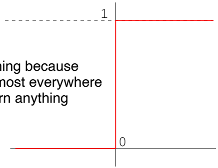
Each neuron (!) implements a logistic regressor

$$\sigma(wx + b)$$

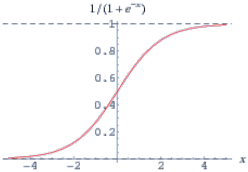
Different activation functions

The activation function is responsible for threshold triggering.

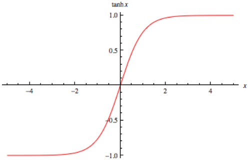
Not good for learning because
the derivative is 0 almost everywhere
so you don't learn anything



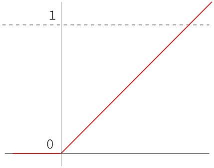
threshold: if $x > 0$ then 1 else 0



logistic function: $\frac{1}{1+e^{-x}}$

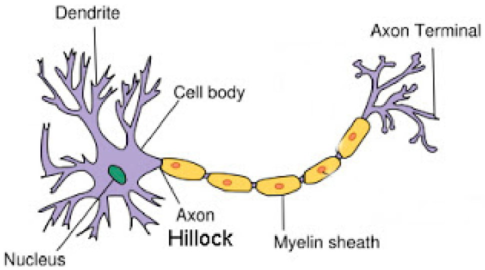


hyperbolic tangent: $\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$



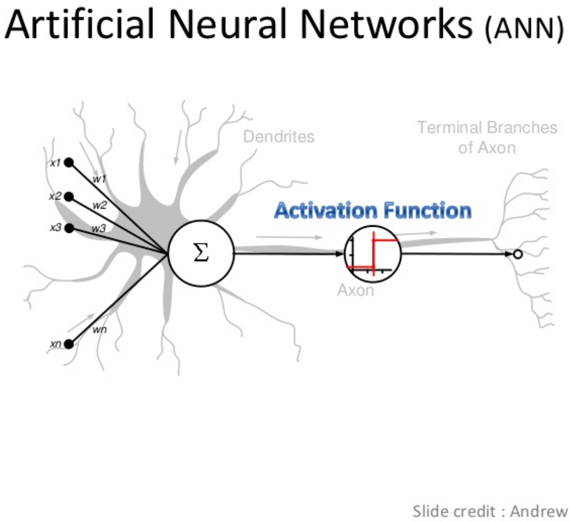
rectified linear (RELU): if $x > 0$ then x else 0

The cortical neuron



- the dendritic tree of the cell collects inputs from other neurons, that get summed together
- when a triggering threshold is exceeded, the Axon Hillock generate an impulse that get transmitted through the axon to other neurons.

A comparison with the cortical neuron



Some figures for human brains

- number of neurons: $\sim 2 \cdot 10^{10}$
- switching time for neuron: $1 - 5$ ms. (slow!)
- synapses (connections) per neuron: $\sim 10^{4-5}$
- time to process an image: 100 ms.

not too deep (< 100)

very high parallelism

Motivations behind neural computation

- to understand, via simulation, how the brain works
- to investigate a different paradigm of computation

  very far from traditional programming languages
- **to solve practical problems difficult to address with algorithmic techniques**

  useful even if the brain works in a different way

Next argument

Network topologies

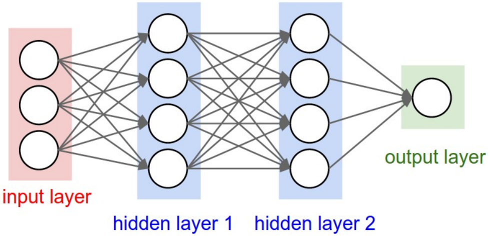
Feed-forward and recurrent networks

If the network is acyclic, it is called a feed-forward network.

If it has cycles it is called recurrent.

Layers

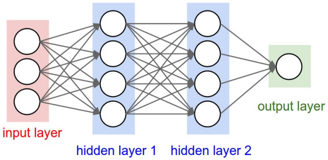
In a feed-forward network, neurons are usually organized in layers.



If there is more than one hidden layer the network is deep, otherwise it is called a shallow network.

Main layers in feed-forward networks: dense layer

Dense layer: each neuron at layer $k-1$ is connected to **each** **each** neuron at layer k.



A single neuron:

$$I^n \cdot W^n + B^1 = O^1$$

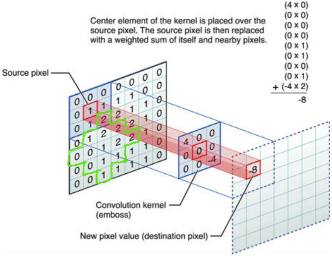
the operation can be vectorized to pruduce m outputs in parallel:

$$I^n \cdot W^{n \times m} + B^m = O^m$$

- dense layers usually work on flat (unstructured) inputs
- the order of elements in input is irrelevant

Main layers in feed-forward networks: convolutional layer

Convolutional layer: each neuron at layer $k-1$ is connected via a parametric **kernel** to a fixed subset of neurons at layer k. The kernel is convolved over the whole previous layer.



1. move the kernel K over a portion M of the input of equal size
2. compute the dot product $M \cdot K$ and possibly add a bias
3. shift the kernel and repeat

The dimension of the output only depends from the number of times the kernel is applied.

Input is structured, and the structure is reflected in the output.

Parameters and hyper-parameters

The weights W_k are the $\boxed{\text{parameters}}$ of the model: they are **learned** during the training phase.

The number of neurons and the way they are connected together are $\boxed{\text{hyper-parameters}}$: they are chosen by the user and **fixed** before training may start.

Other important hyper-parameters govern training such as learning rate, batch-size, number of ephocs an many others.

Features and deep features

Features

Any individual measurable property of data useful for the solution of a specific task is called a feature.

Examples:

- **Medical Diagnosis**: information about the patient (age, clinical history, . . .), symptoms, physichal examination, results of medical tests, . . .
- **Meteo forecasting**: humidity, pression, temperature, wind, rain, snow, ...
- **Image Processing:** raw pixels, combination of adjacent pixels, ...

Signals, Data, Features

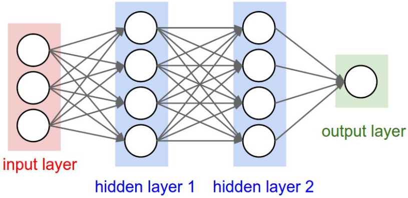
- **Signals**: raw input collected from sensors
- **Data**: meaningful but not focused
- **Features**: meaningful and focused

Deep learning, in deeper sense

Discovering good features is a **complex task**.

Why not delegating the task to the machine, learning them?

Deep learning exploits a *hierarchical organization* of the learning model, allowing complex features to be computed in terms of simpler ones, through non-linear transformations.



Each layer synthesize new features in terms of the previous ones.

AI, machine learning, deep learning

- **Knowledge-based systems**: take an expert, ask him how he solves a problem and try to mimic his approach by means of logical rules

AI, machine learning, deep learning

- **Knowledge-based systems**: take an expert, ask him how he solves a problem and try to mimic his approach by means of logical rules

- **Traditional Machine-Learning**: take an expert, ask him what are the features of data relevant to solve a given problem, and let the machine learn the mapping

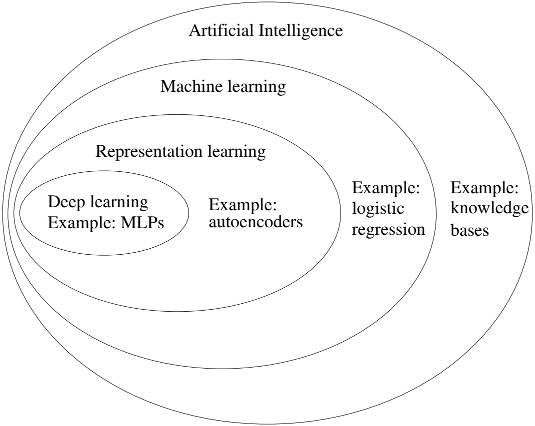
AI, machine learning, deep learning

- **Knowledge-based systems**: take an expert, ask him how he solves a problem and try to mimic his approach by means of logical rules

- **Traditional Machine-Learning**: take an expert, ask him what are the features of data relevant to solve a given problem, and let the machine learn the mapping

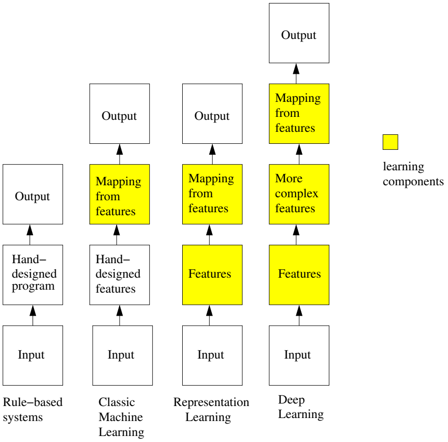
- **Deep-Learning**: get rid of the expert

Relations between research areas



Picture from "Deep Learning" by Y.Bengio, I.Goodfellow e A.Courville, MIT Press.

Components trained to learn



Picture from "Deep Learning" by Y.Bengio, I.Goodfellow e A.Courville, MIT Press.

Next arguments

Diving into DL

A first example

[demo]

Understanding DL

- understand the **different layers**, and their purpose
- understand how layers can be organized in **relevant architectures**
- understand the different possible **applications** of DL, and their specific solutions
- understand the main **issues, problems** and **costs**

Frameworks for DL

- TensorFlow/Keras, Google Brain
- PyTorch, Facebook
- MXNET, Apache

We shall mostly use Keras.

Historical remarks - Legacy

Legacy

1958	perceptron
1975	backpropagation
1980	convolutional layers
1992	Max-pooling
1997	LSTM
...	...

Extremely slow progress

Neural Networks played a marginal role in AI

The Deep Learning revolution

2011	Google Brain foundation	2017	Mask-RCNN
2012	ReLU and Dropout	2017	PPO
2012	ImageNet Competition	2018	Transformers
2013	DQN	2018	BERT/GPT
2014	GANs	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	Inception v1	2020	Vision Transformes
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	MXNet release
2015	AlphaGo	2021	TFP release
2015	Batchnormalization	2022	Keras-CV
2015	YOLO v1	2022	Dall·E 2, Imagen
2015	OpenAI foundation	2022	ChatGPT
2016	Residual connections	2023	LLaMA-2
2017	PyTorch release	2023	ChatGPT-4

Just to mention a few milestones ...

The Deep Learning revolution

Frameworks and Libraries

2011	Google Brain foundation	2017	Mask-RCNN
2012	ReLU and Dropout	2017	PPO
2012	ImageNet Competition	2018	Transformers
2013	DQN	2018	BERT/GPT
2014	GANs	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	Inception v1	2020	Vision Transformes
2015	**Tensorflow release**	2020	OpenAI Jukebox
2015	**Keras release**	2021	**MXNet release**
2015	AlphaGo	2021	**TFP release**
2015	Batchnormalization	2022	**Keras-CV**
2015	YOLO v1	2022	Dall·E 2, Imagen
2015	OpenAI foundation	2022	ChatGPT
2016	Residual connections	2023	LLaMA-2
2017	**PyTorch release**	2023	ChatGPT-4

The Deep Learning revolution

Some popular technical improvements

2011	Google Brain foundation	2017	Mask-RCNN
2012	**ReLU and Dropout**	2017	PPO
2012	ImageNet Competition	2018	Transformers
2013	DQN	2018	BERT/GPT
2014	GANs	2018	Soft Actor Critic
2014	**Attention**	2019	AlphaStar
2014	Inception v1	2020	Vision Transformes
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	MXNet release
2015	AlphaGo	2021	TFP release
2015	**BatchNormalization**	2022	Keras-CV
2015	YOLO v1	2022	Dall·E 2, Imagen
2015	OpenAI foundation	2022	ChatGPT
2016	**Residuality** & Resnet	2023	LLaMA-2
2017	PyTorch release	2023	ChatGPT-4

The Deep Learning revolution

Important Architectures

2011	Google Brain foundation	2017	**Mask-RCNN**
2012	ReLU and Dropout	2017	PPO
2012	ImageNet Competition	2018	**Transformers**
2013	DQN	2018	BERT/GPT
2014	**GANs**	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	**Inception v1**	2020	**Vision Transformes**
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	MXNet release
2015	AlphaGo	2021	TFP release
2015	BatchNormalization	2022	Keras-CV
2015	**YOLO v1**	2022	Dall·E 2, Imagen
2015	OpenAI foundation	2022	ChatGPT
2016	Residuality & **Resnet**	2023	LLaMA-2
2017	PyTorch release	2023	ChatGPT-4

The Deep Learning revolution

Deep Reinforcement Learning

2011	Google Brain foundation	2017	Mask-RCNN
2012	ReLU and Dropout	2017	**PPO**
2012	ImageNet Competition	2018	Transformers
2013	**DQN**	2018	BERT/GPT
2014	GANs	2018	**Soft Actor Critic**
2014	Attention	2019	**AlphaStar**
2014	Inception v1	2020	Vision Transformes
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	MXNet release
2015	**AlphaGo**	2021	TFP release
2015	BatchNormalization	2022	Keras-CV
2015	YOLO v1	2022	Dall·E 2, Imagen
2015	OpenAI foundation	2022	ChatGPT
2016	Residuality & Resnet	2023	LLaMA-2
2017	PyTorch release	2023	ChatGPT-4

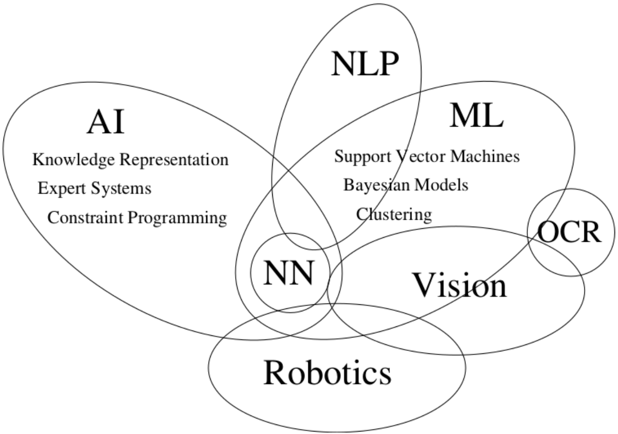
The New Revolution

Generative AI and Large Language Models

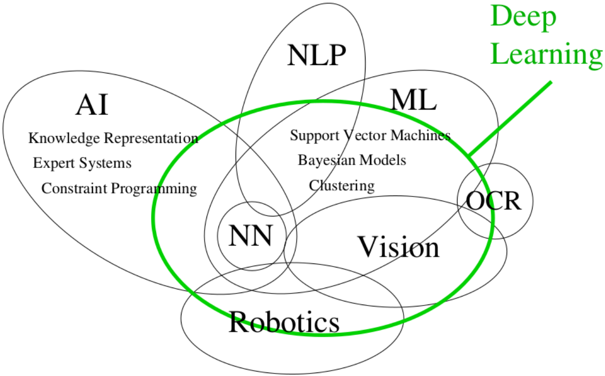
2011	Google Brain foundation
2012	ReLU and Dropout
2012	ImageNet Competition
2013	DQN
2014	GANs
2014	**Attention**
2014	Inception v1
2015	Tensorflow release
2015	Keras release
2015	BatchNormalization
2015	YOLO v1
2015	OpenAI foundation
2016	Residuality & Resnet
2017	PyTorch release

2017	Mask-RCNN
2017	PPO
2018	Transformers
2018	**BERT/GPT**
2018	Soft Actor Critic
2019	AlphaStar
2020	Vision Transformes
2020	**OpenAI Jukebox**
2021	MXNet release
2021	TFP release
2022	Keras-CV
2022	**Dall·E 2, Imagen**
2022	**ChatGPT**
2023	**LLaMA-2**
2023	**ChatGPT-4**

The situation at the beginning of the century



The deep learnig era



See my blog for a short historical perspective.

Didactic Information

- structure of the course
- books, tutorials and blogs
- software
- examination
- office hours

Structure of the course

Frontal lessons intermixed with demos + labs

- Domains of application of Deep Learning
- Expressiveness
- Backpropagation
- Convolutional Networks
- Understanding CNNs
- Object Detection and Segmentation
- Autoencoders
- Generative Adversarial Networks
- Recurrent Networks
- LSTM, Attention, Transformers
- Reinforcement Learning

Text Books

- Dive into deep learning (D2L)
- Y.Bengio, I.Goodfellow and A.Courville. Deep Learning, MIT Press to appear.

Online Tutorials and Blogs

Possible to study on online material (fast updating):

- Tensorflow tutorials
- Towards data science
- Keras blog. By F.Chollet.
- Machine learning tutorial with Python
- Deep Learning Tutorial. LISA lab. University of Montreal.
- a lot of interesting lessons and seminars on youtube
- a lot of material on github
- ...

Code and Datasets

The State of the Art site! (papers with code)

- Tensor flow dataset
- Kaggle Datasets
- Many standard datasets for image processing: Pascal VOC, Coco, . . .
- Face detection and recongition: CelebA, Labeled Faces in the Wild, . . .
- Biomedical challenges
- Amazon Datasets
- ...

Assessment: NEW

At each exam session you will receive a project assignement that you are supposed to complete in **7 days**.

You are supposed to deliver:

- the code source (Keras/TensorFlow) in the form of a **single**, **commented** pyhton notebook

The work will be evaluated according to

1. 80% : comparative evaluation of results (measured in an objective way according to given metrics);
2. 20% : descriptive quality of the notebook

You may possibly integrate the grade with an oral examination.

Office hours

Office hours: On appointment

Prof.Andrea Asperti
andrea.asperti@unibo.it
Via Zanolini 41 - Ex Veneta Station

Tutors:

- Salvatore Fiorilla: salvatore.fiorilla@unibo.it
- Fabio.Merizzi: fabio.merizzi@unibo.it